

Deploy Java Web Application on Docker Using Jenkins on AWS

This project demonstrates deploying a Java web application on a Docker container hosted on an AWS EC2 instance, fully automated using Jenkins CI/CD pipeline integrated with GitHub.

1. Setup Jenkins Server on AWS EC2
2. Setup & Configure Maven and Git
3. Integrate GitHub and Maven with Jenkins
4. Setup Docker Host
5. Integrate Docker with Jenkins
6. Automate Build and Deployment using Jenkins
7. Test the Deployment

Prerequisites:

- AWS Account
- Git/GitHub Account with source code
- Local machine with CLI access
- Familiarity with Docker and Git

Step 1: Setup Jenkins Server on AWS EC2

1. Launch a Linux EC2 instance (t2.micro recommended for free tier).
2. Select existing key pair or create a new one.
3. Configure Security Group:
 - Allow SSH (port 22)
 - Allow HTTP/Jenkins (port 8080)

4. Connect via SSH:

```
bash
```

```
ssh -i new3112.pem ec2-user@<public-ip>
```

5. Install Jenkins:

```
sudo wget -O /etc/yum.repos.d/jenkins.repo https://pkg.jenkins.io/redhat-stable/jenkins.repo
```

```
sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io-2023.key
```

```
sudo yum install epel-release -y
```

```
sudo yum install java-11-openjdk-devel -y
```

```
sudo yum install jenkins -y
```

```
sudo systemctl enable jenkins
```

```
sudo systemctl start jenkins
```

6. Access Jenkins Web UI: http://<EC2_PUBLIC_IP>:8080

7. Unlock Jenkins using:

```
sudo cat /var/lib/jenkins/secrets/initialAdminPassword
```

Step 2: Integrate GitHub with Jenkins

1. Install Git:

```
sudo yum install git -y
```

```
git --version
```

2. Install GitHub plugin in Jenkins (Manage Jenkins → Manage Plugins → Available → GitHub Integration → Install without restart)

3. Configure Git in Jenkins (Manage Jenkins → Global Tool Configuration):

Name: Git

Path: /usr/bin/git

4. Save changes.

Step 3: Integrate Maven with Jenkins

1. Download and install Maven:

```
cd /opt
```

```
wget https://dlcdn.apache.org/maven/maven-3/3.9.5/binaries/apache-maven-3.9.5-bin.tar.gz
```

```
tar -xzf apache-maven-3.9.5-bin.tar.gz
```

2. Set environment variables in : sudo nano /etc/profile.d/maven.sh

```
export JAVA_HOME=/usr/lib/jvm/default-java
```

```
export M2_HOME=/opt/maven
```

```
export MAVEN_HOME=/opt/maven
```

```
export PATH=${M2_HOME}/bin:${PATH}
```

3. Install Maven Integration Plugin in Jenkins

4. Configure Java and Maven paths in Jenkins (Manage Jenkins → Global Tool Configuration)

Step 4: Setup Docker Host

1. Launch another EC2 instance for Docker (or use same if preferred)

2. Install Docker:

```
sudo yum install docker -y
```

```
sudo systemctl enable docker
```

```
sudo systemctl start docker
```

```
docker --version
```

3. Run basic Tomcat container:

```
docker run -d --name tomcat-container -p 8081:8080 tomcat
```

4. Copy content from webapps.dist to webapps:

```
docker exec -it tomcat-container /bin/bash
```

```
cp -R /usr/local/tomcat/webapps.dist/* /usr/local/tomcat/webapps
```

```
exit
```

5.Create custom Dockerfile:

```
FROM tomcat:latest
```

```
RUN cp -R /usr/local/tomcat/webapps.dist/* /usr/local/tomcat/webapps
```

Step 5: Integrate Docker with Jenkins

1.Create dockeradmin user and add to docker group:

```
sudo useradd dockeradmin
```

```
sudo passwd dockeradmin
```

```
sudo usermod -aG docker dockeradmin
```

2.Enable password authentication in /etc/ssh/sshd_config:

```
PasswordAuthentication yes
```

3.Reload SSH service:

```
sudo systemctl reload sshd
```

4.Install "Publish Over SSH" plugin in Jenkins (Manage Jenkins → Manage Plugins)

5.Add Docker host SSH credentials and configure Publish Over SSH.

Step 6: Jenkins Job to Build & Deploy to Docker

1.Create Jenkins job → configure to Poll SCM

2.Use Send files or execute commands over SSH post-build step

3.Example commands to build and run Docker container:

```
cd /opt/docker
```

```
docker build -t regapp:v1 .
```

```
docker run -d --name registerapp -p 8087:8080 regapp:v1
```

4.Trigger job automatically on GitHub commits.

Step 7: Update Dockerfile to Include WAR

```
FROM tomcat:latest  
RUN cp -R /usr/local/tomcat/webapps.dist/* /usr/local/tomcat/webapps  
COPY ./*.war /usr/local/tomcat/webapps
```

Build and run new image:

```
docker build -t tomcat:v1 .  
docker run -d --name tomcatv1 -p 8086:8080 tomcat:v1
```

Access application: http://<DOCKERHOST_PUBLIC_IP>:8086/webapp/

Step 8: Automate End-to-End Deployment

Commit changes to GitHub → triggers Jenkins build → copies WAR to Docker host → builds Docker image → runs container

Example Exec command in Jenkins post-build:

```
cd /opt/docker  
docker build -t regapp:v1 .  
docker run -d --name registerapp -p 8087:8080 regapp:v1
```

Access deployed app: http://<DOCKERHOST_PUBLIC_IP>:8087/webapp/

Conclusion

This project automates the full CI/CD pipeline for a Java web application using GitHub, Jenkins, Docker, and AWS EC2.

Output:

Ec2:

<input checked="" type="checkbox"/> jenkins-git	i-0494ddc6e0718ed3c	Running	Q	t3.micro	3/3 checks passed	View alarms +	us-west-2b	ec2-34-221-
<input checked="" type="checkbox"/> docker	i-0eb83e6aa0e1b98f5	Running	Q	t3.micro	3/3 checks passed	View alarms +	us-west-2b	ec2-44-246

Jdk & mvn :

```
No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-34-2:~$ java -version
openjdk version "21.0.9" 2025-10-21
OpenJDK Runtime Environment (build 21.0.9+10-Ubuntu-124.04)
OpenJDK 64-Bit Server VM (build 21.0.9+10-Ubuntu-124.04, mixed mode, sharing)
ubuntu@ip-172-31-34-2:~$ wget https://dlcdn.apache.org/maven/maven-3/3.9.12/binaries/apache-maven-3.9.12-bin.tar.gz -P /tmp
--2025-12-31 06:26:53 - https://dlcdn.apache.org/maven/maven-3/3.9.12/binaries/apache-maven-3.9.12-bin.tar.gz
Resolving dlcdn.apache.org (dlcdn.apache.org)... 151.101.2.132, 2a04:4e42:644
Connecting to dlcdn.apache.org (dlcdn.apache.org)|151.101.2.132|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 9233336 (8.8M) [application/x-gzip]
Saving to: '/tmp/apache-maven-3.9.12-bin.tar.gz'

apache-maven-3.9.12-bin.tar.gz      100%[=====] 8.80M --.-KB/s   in 0.09s

2025-12-31 06:26:53 (101 MB/s) - '/tmp/apache-maven-3.9.12-bin.tar.gz' saved [9233336/9233336]

ubuntu@ip-172-31-34-2:~$ sudo tar xf /tmp/apache-maven-*.tar.gz -C /opt

ubuntu@ip-172-31-34-2:~$ sudo chmod +x /etc/profile.d/maven.sh
ubuntu@ip-172-31-34-2:~$ source /etc/profile.d/maven.sh
ubuntu@ip-172-31-34-2:~$ mvn -version
Apache Maven 3.9.12 (848fb64bf2d427b72bdb2471c22fcfed7ebd9a7a1)
Maven home: /opt/maven
Java version: 21.0.9, vendor: Ubuntu, runtime: /usr/lib/jvm/java-21-openjdk-amd64
Default locale: en, platform encoding: UTF-8
OS name: "Linux", version: "6.14.0-1015-aws", arch: "amd64", family: "unix"
ubuntu@ip-172-31-34-2:~$ |
```

Jdk & mvn Environmental path:

```
GNU nano 7.2
export JAVA_HOME=/usr/lib/jvm/default-java
export M2_HOME=/opt/maven
export MAVEN_HOME=/opt/maven
export PATH=${M2_HOME}/bin:${PATH}
```

Jenkins:

```
ubuntu@ip-172-31-29-182:~$ sudo systemctl status jenkins.service
● jenkins.service - Jenkins Continuous Integration Server
   Loaded: loaded (/usr/lib/systemd/system/jenkins.service; enabled; preset: enabled)
   Active: active (running) since Fri 2026-01-09 07:45:29 UTC; 3min 40s ago
     Main PID: 543 (java)
        Tasks: 37 (limit: 1017)
       Memory: 394.2M (peak: 422.7M)
          CPU: 28.996s
        CGroup: /system.slice/jenkins.service
           └─543 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=/var/cache/jenkins/war --httpPort=8080

Jan 09 07:45:26 ip-172-31-29-182 jenkins[543]: 2026-01-09 07:45:26.780+0000 [id=31]      INFO      jenkins.InitReactorRunner$1#onAttained: Started all >
Jan 09 07:45:26 ip-172-31-29-182 jenkins[543]: 2026-01-09 07:45:26.798+0000 [id=35]      INFO      jenkins.InitReactorRunner$1#onAttained: Augmented al>
Jan 09 07:45:27 ip-172-31-29-182 jenkins[543]: 2026-01-09 07:45:27.683+0000 [id=31]      INFO      h.p.b.g.GlobalTimeOutConfiguration$load: global time>
Jan 09 07:45:29 ip-172-31-29-182 jenkins[543]: 2026-01-09 07:45:29.599+0000 [id=34]      INFO      jenkins.InitReactorRunner$1#onAttained: System config>
Jan 09 07:45:29 ip-172-31-29-182 jenkins[543]: 2026-01-09 07:45:29.600+0000 [id=34]      INFO      jenkins.InitReactorRunner$1#onAttained: System config>
Jan 09 07:45:29 ip-172-31-29-182 jenkins[543]: 2026-01-09 07:45:29.744+0000 [id=30]      INFO      jenkins.InitReactorRunner$1#onAttained: Loaded all jo>
Jan 09 07:45:29 ip-172-31-29-182 jenkins[543]: 2026-01-09 07:45:29.748+0000 [id=35]      INFO      jenkins.InitReactorRunner$1#onAttained: Configuration>
Jan 09 07:45:29 ip-172-31-29-182 jenkins[543]: 2026-01-09 07:45:29.810+0000 [id=30]      INFO      jenkins.InitReactorRunner$1#onAttained: Completed in>
Jan 09 07:45:29 ip-172-31-29-182 jenkins[543]: 2026-01-09 07:45:29.847+0000 [id=24]      INFO      hudson.lifecycle.Lifecycle#onReady: Jenkins is fully&
```

Getting Started

Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log ([not sure where to find it?](#)) and this file on the server:

`/var/lib/jenkins/secrets/initialAdminPassword`

Please copy the password from either location and paste it below.

Administrator password

.....

Continue

Jdk & mvn path:

JDK installations

+ Add JDK

⋮ JDK

Name

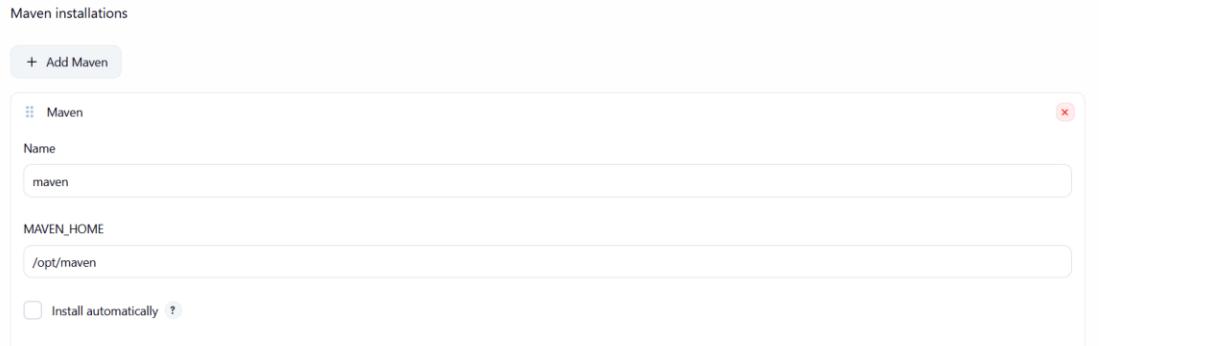
java

x

JAVA_HOME

/usr/lib/jvm/java-21-openjdk-amd64

Install automatically ?



New Jenkins job:

DevOps-Project-01

New Project

Permalinks

- Last build (#3), 58 min ago
- Last stable build (#3), 58 min ago
- Last successful build (#3), 58 min ago
- Last completed build (#3), 58 min ago

Builds > ...

Filter

Today

#3 6:05 AM

Docker:

```
ubuntu@ip-172-31-29-137:~$ docker info
Client: Docker Engine - Community
  Version: 29.1.3
  Context: default
  Debug Mode: false
  Plugins:
    buildx: Docker Buildx (Docker Inc.)
      Version: v0.30.1
      Path: /usr/libexec/docker/cli-plugins/docker-buildx
    compose: Docker Compose (Docker Inc.)
      Version: v5.0.1
      Path: /usr/libexec/docker/cli-plugins/docker-compose

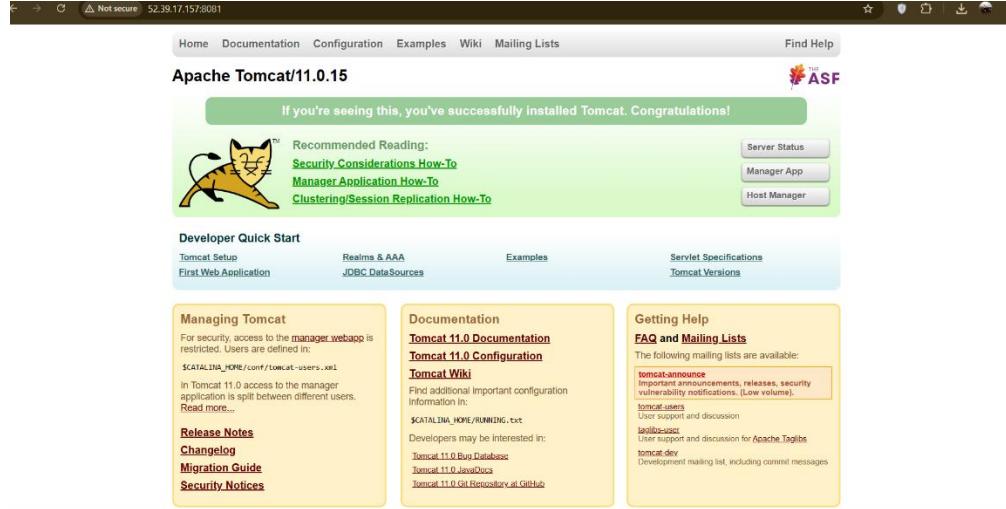
Server:
  permission denied while trying to connect to the docker API at unix:///var/run/docker.sock
ubuntu@ip-172-31-29-137:~$ |
```

```
ubuntu@ip-172-31-29-137:~$ sudo su
root@ip-172-31-29-137:/home/ubuntu# cd ~
root@ip-172-31-29-137:~# docker ps
CONTAINER ID   IMAGE       COMMAND           CREATED          STATUS          PORTS          NAMES
a776a1ce0025   tomcat:v1   "catalina.sh run"   53 minutes ago   Up 53 minutes   0.0.0.0:8086->8080/tcp, [::]:8086->8080/tcp   tomcatv1
d65dfa09468b   1b6ale45687e  "catalina.sh run"   2 hours ago     Up 2 hours     0.0.0.0:8087->8080/tcp, [::]:8087->8080/tcp   myapp
root@ip-172-31-29-137:~# |
```

```
root@ip-172-31-29-137:~# docker images
```

IMAGE	ID	DISK USAGE	CONTENT SIZE	EXTRA
myapp:v1	94c31b175f3b	595MB	156MB	U
tomcat:latest	689475f36b76	587MB	157MB	U
tomcat:v1	152e9e8d29e8	595MB	156MB	U
tomcatserver:latest	4657211b6655	595MB	156MB	U

Tomcat :



The screenshot shows the Apache Tomcat 11.0.15 web interface. At the top, there's a navigation bar with links to Home, Documentation, Configuration, Examples, Wiki, and Mailing Lists. A "Find Help" search bar is also present. The main content area features a green banner with the text "If you're seeing this, you've successfully installed Tomcat. Congratulations!" and a cartoon cat icon. Below the banner are several links: Recommended Reading (Security Considerations How-To, Manager Application How-To, Clustering/Session Replication How-To), Server Status, Manager App, and Host Manager. The "Developer Quick Start" section includes links to Tomcat Setup, First Web Application, Realms & AAA, JDBC DataSources, Examples, and Servlet Specifications. The "Documentation" section links to Tomcat 11.0 Documentation, Tomcat 11.0 Configuration, and Tomcat Wiki. The "Getting Help" section links to FAQ and Mailing Lists, listing tomcat-announce, tomcat-users, tomcat-dev, tomcat-javadoc, and tomcat-svn.

User :

dockeradmin

```
dockeradmin:x:1001:1001:Jagadish,5,9944258534,9944258534,other:/home/dockeradmin:/bin/bash
root@ip-172-31-29-137:/etc#
```

Now in order to access the server using the newly created User we need to allow password-based authentication. For that, we need to do some changes in the /etc/ssh/sshd_config file.

```
root@ip-172-31-29-137:~# vi /etc/ssh/sshd_config file.
```

```
# To disable tunneled clear text passwords, change to no here!
PasswordAuthentication yes
PubkeyAuthentication yes
KbdInteractiveAuthentication yes
ChallengeResponseAuthentication yes
UsePAM yes
#PermitEmptyPasswords no
```

Publish over SSH

SSH Servers

SSH Server
Name ? docker
Hostname ? 172.31.29.137
Username ? dockeradmin
Remote Directory ? /home/dockeradmin
<input type="checkbox"/> Avoid sending files that have not changed ?

Advanced  Edit

Jenkins configure:

The screenshot shows the Jenkins configuration interface for a Git project. On the left, there's a sidebar with links: Source Code Management (selected), Triggers, Environment, Build Steps, and Post-build Actions. The main area is titled 'Git' and shows a 'Repositories' section. It includes fields for 'Repository URL' (set to 'https://github.com/Jagadishsrvn/DevOps-Project.git'), 'Credentials' (set to '- none -'), an 'Advanced' dropdown, and a '+ Add Repository' button. Below this is a 'Branches to build' section with a single branch listed.

Triggers

Set up automated actions that start your build based on specific events, like code changes or scheduled times.

- Trigger builds remotely (e.g., from scripts) ?
- Build after other projects are built ?
- Build periodically ?
- GitHub hook trigger for GITScm polling ?
- Poll SCM ?

Post-build Actions

Define what happens after a build completes, like sending notifications, archiving artifacts, or triggering other jobs.

The screenshot shows the Jenkins configuration interface for post-build actions. It starts with an 'SSH Publishers' section, which includes an 'SSH Server' configuration with a 'Name' set to 'docker'. Below this is an 'Advanced' dropdown. The next section is 'Transfers', which contains a 'Transfer Set' configuration. It includes a 'Source files' field containing 'Project-1/hello-world/webapp/target/*.war' and a 'Remove prefix' field containing 'Project-1/hello-world/webapp/target'.

Docker file:

```
root@ip-172-31-29-137:/opt/docker# cat Dockerfile
FROM tomcat:9.0
RUN cp -R /usr/local/tomcat/webapps.dist/* /usr/local/tomcat/webapps
COPY ./*.war /usr/local/tomcat/webapps/
root@ip-172-31-29-137:/opt/docker# |
```

```
dockeradmin@ip-172-31-29-137:/opt/docker$ ls
Dockerfile  webapp.war
dockeradmin@ip-172-31-29-137:/opt/docker$ |
```

← ⏪ Not secure 44.246.228.119:8086/webapp/

New user Register for DevOps Learning

Please fill in this form to create an account.

Enter Name Enter Full Name
Enter mobile Enter mobile number
Enter Email Enter Email
Password Enter Password
Repeat Password Repeat Password

By creating an account you agree to our [Terms & Privacy](#).

Already have an account? [Sign in](#).

← ⏪ 44.246.228.119:8087/webapp/

New user Register for DevOps Learning

Please fill in this form to create an account.

Enter Name
Enter mobile
Enter Email
Password
Repeat Password

By creating an account you agree to our [Terms & Privacy](#).

Already have an account? [Sign in](#).